REMARKS/ARGUMENTS

In the Office Action, the Examiner noted that claims 1-25 are pending in the application. The Examiner additionally stated that claims 1-25 are rejected. By this communication, claims 1, 8, 15-16, and 21 are amended. Hence, claims 1-25 are pending in the application.

Applicant hereby requests further examination and reconsideration of the application, in view of the foregoing amendments.

In the Specification

Applicant has amended the specification to secure a substantial correspondence between the claims amended herein and the remainder of the specification. No new matter is presented.

In the Claims

Objections

The Examiner noted that Applicant's argument with regards to the objection to claim 15 is not persuasive because claim 15 as amended on 04 February 2008 does not match claim 15 as presently identified as being "Previously Presented," stating that currently claim 15 recites, "wherein said cryptography unit executes said plurality of cryptographic rounds on each of a plurality of input text blocks to generate a corresponding each of a plurality of output text blocks, wherein said plurality of cryptographic rounds are prescribed by a control word that is provided to said cryptography unit," while claim 15 as it was amended on 04 February 2008 recited, "wherein said cryptography unit executes said plurality of cryptographic rounds on each of a plurality of input text blocks to generate a corresponding each of a plurality of output text blocks, wherein said plurality of cryptographic rounds are prescribed by a control word that is provided to said cryptography unit, execution logic comprises: of a plurality of output text blocks, wherein said plurality of cryptographic cryptography unit."

The Examiner pointed out that clearly these two claims are not the same, and present claim 15 has not been identified as being amended, and therefore, the amendments made

to claim 15 on 04 February 2008 were not properly identified. Clarification was requested.

In that the Examiner has noted that the amendment to claim 15 was not properly identified, Applicant thus assumes that 1) the objection to the claim was not withdrawn and 2) the language of claim 15 stands as quoted by the Examiner above. Accordingly, Applicant has amended claim 15 and identified the claim as "Currently Amended." Thus, it is requested that the objection to claim 15 be withdrawn.

Rejections Under 35 U.S.C. §103(a)

The Examiner rejected claims 1-25 under 35 U.S.C. 103(a) as being unpatentable over Kessler, US6789147 (hereinafter, "Kessler"), in view of Colavin, U.S. Publication No. 20040103263 (hereinafter, "Colavin"), and further in view of Miller, US6081884 (hereinafter, "Miller"). Applicant respectfully traverses the Examiner's rejections.

Referring to claims 1 and 21, the Examiner noted that Kessler discloses a co-processor that includes multiple execution units (Figure 2) wherein each of the execution units includes an execution queue to store cryptographic instructions received by the co-processor (Figure 8). The Examiner noted that this meets the limitation of a fetch logic, disposed within a microprocessor, configured to receive a cryptographic instruction as a part of an instruction flow executing on said microprocessor, wherein said cryptographic instruction prescribes one of the cryptographic operations.

The Examiner also stated that the execution units include a plurality of operation blocks that correspond to different cryptographic operations that are used depending upon the type of instruction received in the execution queue (Figure 8 & Col. 9, lines 7-43), which meets the limitation of wherein said cryptographic instruction prescribes one of a plurality of cryptographic algorithms, algorithm logic, operatively coupled to said cryptographic instruction, configured to direct said microprocessor to execute said one of the cryptographic operations according to said one of a plurality of cryptographic algorithms.

The Examiner further observed that using the appropriate operation block, the corresponding cryptographic algorithm is used when processing the received instruction

(Col. 9, lines 28-43), which meets the limitation of execution logic, operatively coupled to said algorithm logic, configured to execute said one of the cryptographic operations.

The Examiner also noted that The operation blocks correspond to cryptographic algorithms such as AES, 3DES, DES, and RC4 (Figures 5 & 8), which meets the limitation of executing a plurality of cryptographic rounds required to complete said one of the cryptographic operations.

The Examiner noted that Kessler does not specify that the co-processor executes that program that includes the cryptographic operations, but that Colavin discloses a host and co-processor configuration wherein the co-processor executes the actual application program (Abstract & [0018]), which meets the limitation of said cryptographic instruction is one of the instructions in an application program, wherein said application is executed by said microprocessor to obtain expected results. The Examiner observed that it would have been obvious to one of ordinary skill in the art at the time the invention was made for the co-processor of Kessler to execute the actual application program as described by Colavin in order to efficiently execute programs with high instruction level parallelism as taught by Colavin ([0002]).

The Examiner stated that Kessler does not specify that the co-processor utilizes the x86 instruction set, however, it would have been obvious to one of ordinary skill in the art at the time the invention was made for the co-processor described in Kessler to implement the x86 instruction set because the x86 instruction set has been widely accepted because of it's compatibility with a large amount of software as taught by Miller (Col. 2, lines 9-14). The Examiner added that Applicant's specification shows that integer instructions are inherent to the x86 instruction set (Page 27), and therefore, when implementing the x86 instruction set in the co-processor of Kessler, as previously described, the execution units would effectively operate as a "integer unit" as claimed.

In reply to Applicant's arguments and remarks made in the previous communication, the Examiner stated that Applicant's attempt to correlate the definition of a microprocessor to fruit is completely off base and irrelevant to the issues.

Applicant responds that the correlation of a microprocessor to fruit was submitted to provide an analogous example of how one skilled in the art would view the definition of a microprocessor versus a co-processor. It is evident, however, from the Examiner's comments, that such an example was not appreciated.

The Examiner noted that Applicant repeatedly makes the argument that a co-processor cannot be considered a "microprocessor," however, the Examiner pointed out that Applicant admits on page 14 of the remarks that a co-processor is in fact a "microprocessor." The Examiner added that Applicant states, "it is readily concurred with by those in the art that a 'coprocessor' is a 'limited microprocessor,'" and that Applicant clearly recognizes the fact that a coprocessor is equivalent to a microprocessor.

The Examiner stated that Applicant argues, "a coprocessor only executes threads, or single instructions, or single tasks, handed off by a host microprocessor...a microprocessor (i.e. CPU) executes application programs." The Examiner noted that while he rejects the contention that a co-processor "only executes threads, or single instructions", a new ground of rejection is provided to address the amended claim language, noting that Colavin (US Publication 2004/0103263) shows a host/co-processor configuration where the co-processor performs execution of application programs.

The Examiner noted that definitions from Wikipedia.com will not be considered evidence since Wikipedia.com is not a reliable source of information.

The Examiner noted that Applicant argues that the motivation to combine Kessler and Miller is "ludicrous and unfounded. Such an observation that the functionality of literally hundreds of extant instructions be added to a specialized encryption coprocessor that only possesses a few primitive functions such as Kessler's is not a reasonable conclusion to make whatsoever." The Examiner pointed out that this argument is not persuasive because Applicant has failed to provide any evidence that the coprocessor of Kessler "possesses a few primitive functions," and that Applicant's contention amounts to mere conclusory statements based on personal opinion. The Examiner stated that Applicant's conclusion that the proposed modification is "ludicrous and unfounded" is not based on any actual technical evidence.

Applicant responds that he is convinced that further arguments directed toward distinguishing a microprocessor from a co-processor would be futile, because it is clear that the Examiner is of the opinion that a co-processor is one species of a microprocessor, as would a graphics controller be another species of a microprocessor.

Applicant also respectfully submits that it is his sincere desire to forward this case through the Office in all candor and good faith, and thus he has amended claims 1, 16, and 21 to recite, among other features and limitations, "an x86-compatible microprocessor." This element is clearly defined in the specification, one instance of such is as is stated in paragraph [0044], which is repeated below for ease of reference. To wit (with emphasis provided by underlining):

> **[0044]** Referring to FIGURE 3, a block diagram 300 is provided featuring a microprocessor apparatus according to the present invention for performing cryptographic operations. The block diagram 300 depicts a microprocessor 301 that is coupled to a system memory 321 via a memory bus 319. The microprocessor 301 includes translation logic 303 that receives instructions from an instruction register 302. The translation logic 303 comprises logic, circuits, devices, or microcode (i.e., micro instructions or native instructions), or a combination of logic, circuits, devices, or microcode, or equivalent elements that are employed to translate instructions into associated sequences of micro instructions. The elements employed to perform translation within the translation logic 303 may be shared with other circuits, microcode, etc., that are employed to perform other functions within the microprocessor 301. According to the scope of the present application, microcode is a term employed to refer to one or more micro instructions. A micro instruction (also referred to as a native instruction) is an instruction at the level that a unit executes. For example, micro instructions are directly executed by a reduced instruction set computer (RISC) microprocessor. For a complex instruction set computer (CISC) microprocessor such as an x86-compatible microprocessor, x86 instructions are translated into associated micro instructions, and the associated micro instructions are directly executed by a unit or units within the CISC microprocessor. The translation logic

303 is coupled to a micro instruction queue 304. The micro instruction queue 304 has a plurality of micro instruction entries 305, 306. Micro instructions are provided from the micro instruction queue 304 to register stage logic that includes a register file 307. The register file 307 has a plurality of registers 308-313 whose contents are established prior to performing a prescribed cryptographic operation. Registers 308-312 point to corresponding locations 323-327 in memory 321 that contain data which is required to perform the prescribed cryptographic operation. The register stage is coupled to load logic 314, which interfaces to a data cache 315 for retrieval of data for performance of the prescribed cryptographic operation. The data cache 315 is coupled to the memory 321 via the memory bus 319. Execution logic 328 is coupled to the load logic 314 and executes the operations prescribed by micro instructions as passed down from previous stages. The execution logic 328 comprises logic, circuits, devices, or microcode (i.e., micro instructions or native instructions), or a combination of logic, circuits, devices, or microcode, or equivalent elements that are employed to perform operations as prescribed by instructions provided thereto. The elements employed to perform the operations within the execution logic 328 may be shared with other circuits, microcode, etc., that are employed to perform other functions within the microprocessor 301. The execution logic 328 includes a cryptography unit 316. The cryptography unit 316 receives data required to perform the prescribed cryptographic operation from the load logic 314. Micro instructions direct the cryptography unit 316 to perform the prescribed cryptographic operation on a plurality of blocks of input text 326 to generate a corresponding plurality of blocks of output text 327. The cryptography unit 316 comprises logic, circuits, devices, or microcode (i.e., micro instructions or native instructions), or a combination of logic, circuits, devices, or microcode, or equivalent elements that are employed to perform cryptographic operations. The elements employed to perform the cryptographic operations within the cryptography unit 316 may be shared with other circuits, microcode, etc., that are employed to perform other functions within the microprocessor 301. In one embodiment, the cryptography

unit 316 operates in parallel to other execution units (not shown) within the execution logic 328 such as an integer unit, floating point unit, etc. One embodiment of a "unit" within the scope of the present application comprises logic, circuits, devices, or microcode (i.e., micro instructions or native instructions), or a combination of logic, circuits, devices, or microcode, or equivalent elements that are employed to perform specified functions or specified operations. The elements employed to perform the specified functions or specified operations within a particular unit may be shared with other circuits, microcode, etc., that are employed to perform other functions or operations within the microprocessor 301. For example, in one embodiment, an integer unit comprises logic, circuits, devices, or microcode (i.e., micro instructions or native instructions), or a combination of logic, circuits, devices, or microcode, or equivalent elements that are employed to execute integer instructions. A floating point unit comprises logic, circuits, devices, or microcode (i.e., micro instructions or native instructions), or a combination of logic, circuits, devices, or microcode, or equivalent elements that are employed to execute floating point instructions. The elements employed execute integer instructions within the integer unit may be shared with other circuits, microcode, etc., that are employed to execute floating point instructions within the floating point unit. In one embodiment that is compatible with the x86 architecture, the cryptography unit 316 operates in parallel with an x86 integer unit, an x86 floating point unit, an x86 MMX® unit, and an x86 SSE® unit. According to the scope of the present application, an embodiment is compatible with the x86 architecture if the embodiment can correctly execute a majority of the application programs that are designed to be executed on an x86 microprocessor. An application program is correctly executed if its expected results are obtained. Alternative x86-compatible embodiments contemplate the cryptography unit operating in parallel with a subset of the aforementioned x86 execution units. The cryptography unit 316 is coupled to store logic 317 and provides the corresponding plurality of blocks of output text 327. The store logic 317 is also coupled to the data cache 315, which routes the

> output text data 327 to system memory 321 for storage. The store logic 317 is coupled to write back logic 318. The write back logic 318 updates registers 308-313 within the register file 307 as the prescribed cryptographic operation is accomplished. In one embodiment, micro instructions flow through each of the aforementioned logic stages 302, 303, 304, 307, 314, 316-318 in synchronization with a clock signal (not shown) so that operations can be concurrently executed in a manner substantially similar to operations performed on an assembly line.

Thus, recitation in the independent claims of "an x86-compatible microprocessor" is submitted to overcome the rejections noted above. By way of summary, in order for any device to be deemed equivalent to an x86-compatible microprocessor as defined above, it must be 1) compatible with the x86 architecture, 2) it must include an x86 integer unit, 3) it must include an x86 floating point unit, 4) it must include an x86 MMX unit, and 5) it must include an x86 SSE unit. In addition, an equivalent device must 6) correctly execute a majority of the application programs that are designed to be executed on an x86 microprocessor.

Applicant has thoroughly studied the teachings of Kessler, Colavin, and Miller, both alone and in combination, and finds that Kessler and Colavin fail to teach any form of an x86-compatible microprocessor. As has been previously submitted, Kessler teaches a security co-processor interface. He does not teach an x86 integer unit, x86 floating point unit, x86 MMX unit, or x86 SSE unit. Kessler fails to teach or suggest the capability to correctly execute a majority of the application programs that are designed to be executed on an x86 microprocessor.

Colavin teaches clustered VLIW processing elements, coupled by a runtime reconfigurable inter-cluster interconnect to form a coprocessor executing only those portions of a program having high instruction level parallelism. (Abstract). Applicant respectfully submits that Colavin clearly states in this brief sentence that his technique is not capable of the execution of application programs, but *only those portions of a program having high instruction level parallelism.* Thus, Applicant respectfully asserts that the Examiner's conclusion Colavin's co-processor executes the actual application

program, which meets the limitation of said cryptographic instruction is one of the instructions in an application program, wherein said application is executed by said microprocessor to obtain expected results, is refuted by the above excerpt from Colavin. Furthermore, Colavin does not teach an x86 integer unit, x86 floating point unit, x86 MMX unit, or x86 SSE unit. In addition, Colavin fails to teach or suggest the capability to correctly execute a majority of the application programs that are designed to be executed on an x86 microprocessor.

Regarding the teachings of Miller, Applicant indeed agrees that the x86 instruction set has been widely accepted because of it's compatibility with a large amount of software. However, Miller in no way suggests that a microprocessor be provided that is both x86 compatible and that has an x86 cryptographic instruction that takes advantage of an integral cryptographic unit. Rather, Miller's motivation is to provide an alignment unit within a microprocessor that is configured to detect variable length instructions as they are fetched from an instruction cache and then embed the variable length instruction within a long instruction word, thus achieving the benefits of a long instruction word format, while still retaining backward compatibility with variable length instructions. Miller's microprocessor is indeed configured according to some aspects of the x86 architecture, but he fails to teach or suggest a cryptographic instruction for execution, or furthermore, any execution unit that may be capable of performing a cryptographic operation.

Applicant respectfully submits that neither Kessler, Colavin, nor Miller teach an x86-compatible microprocessor that executes cryptographic instruction as part of an application program, and that performs a specified cryptographic operation. In addition, it is respectfully asserted that none of the noted references would point one skilled in the art to provide a cryptographic instruction that can be executed by an x86-compatible microprocessor.

By combining the two references, one skilled in the art would be led to conclude that Kessler's coprocessor may be useful in an x86 environment because it could offload cryptographic functions which would otherwise have to be performed via operating

system intensive subroutine calls, and that Colavin's technique may be useful for those portions of a program having high instruction level parallelism.

Based upon the above arguments and instant amendments, Applicant respectfully requests that the rejection of claim 1 be withdrawn.

Claim 21 recites substantially the same limitations as have been argued above as being allowable over Kessler, Colavin, Miller, or a combination of these references. Accordingly, it is requested that the rejection of claim 21 be withdrawn as well

With respect to claims 2-15, these claims depend from claim 1 and add further limitations that are neither anticipated nor made obvious by Kessler, Colavin, Miller, or a combination of these references. Accordingly, Applicant respectfully requests that the Examiner withdraw the rejections of claims 2-15.

With respect to claims 22-25, these claims depend from claim 21 and add further limitations that are neither anticipated nor made obvious by Kessler, Colavin, Miller, or a combination of these references. Accordingly, Applicant respectfully requests that the Examiner withdraw the rejections of claims 22-25.

As per claim 16, the Examiner noted that Kessler discloses a co-processor that includes multiple execution units (Figure 2) wherein each of the execution units includes an execution queue to store cryptographic instructions received by the co-processor (Figure 8), which meets the limitation of a cryptographic unit within a microprocessor, configured to execute one of the cryptographic operations response to receipt of a cryptographic instruction that prescribes said one of the cryptographic operations, wherein said cryptographic instruction is one of the instructions in an application program that are fetched from memory by fetch logic in said microprocessor. The Examiner additionally observed that the execution units include a plurality of operation blocks that correspond to different cryptographic operations that are used depending upon the type of instruction received in the execution queue (Figure 8 & Col. 9, lines 7-43), which meets the limitation of an algorithm field, configured to prescribed one of a plurality of cryptographic algorithms to be employed when executing said one of the cryptographic operations. The Examiner noted that using the appropriate operation

block, the corresponding cryptographic algorithm is used when processing the received instruction (Col. 9, lines 28-43), which meets the limitation of algorithm logic, operatively coupled to said cryptography unit, configured to direct said device to perform said one of the cryptographic operations according to said one of the plurality of cryptographic algorithms.

The Examiner also noted that Kessler does not specify that the co-processor executes that program that includes the cryptographic operations, but that Colavin discloses a host and co-processor configuration wherein the co-processor executes the actual application program (Abstract & [0018]), which meets the limitation of wherein said microprocessor executes said application program to obtain expected results. The Examiner then observed that it would have been obvious to one of ordinary skill in the art at the time the invention was made for the co-processor of Kessler to execute the actual application program as described by Colavin in order to efficiently execute programs with high instruction level parallelism as taught by Colavin ([0002]).

The Examiner further noted that Kessler does not specify that the coprocessor utilizes the x86 instruction set, however, it would have been obvious to one of ordinary skill in the art at the time the invention was made for the co-processor described in Kessler to implement the x86 instruction set because the x86 instruction set has been widely accepted because of it's compatibility with a large amount of software as taught by Miller (Col. 2, lines 9-14). The Examiner added that Applicant's specification shows that integer instructions are inherent to the x86 instruction set (Page 27) and, therefore, when implementing the x86 instruction set in the coprocessor of Kessler, as previously described, the execution units would effectively operate as a "integer unit" as claimed.

Applicant respectfully disagrees and directs the Examiner's attention to arguments provided above in traversal of the rejections of claims 1 and 21. More specifically, claim 16, as amended herein, recites, an x86-compatible microprocessor that, among other features and limitations, has a cryptography unit, configured to execute one of the cryptographic operations responsive to receipt of a cryptographic instruction that prescribes said one of the cryptographic operations, wherein said cryptographic

instruction is one of the instructions in an application program that are fetched from memory by fetch logic in said x86-compatible microprocessor, and wherein said x86-compatible microprocessor executes said application program. As has been argued above, Kessler's coprocessor is incapable of executing an application program, Colavin's mechanism only executes only those portions of a program having high instruction level parallelism. None of these references teach any of the x86 units that are resident in an x86-compatible microprocessor.

As noted above, Miller indeed teaches a technique for aligning x86 instructions along with RISC instructions. But Kessler, Colavin, and Miller, utterly fail to teach a cryptographic instruction that can be executed on an x86-compatible microprocessor as part of an application program.

Since these limitations are not taught, contemplated, or suggested by Kessler, Colavin, Miller, or a combination of these references, it is requested that the rejection of claim 16 be withdrawn.

With respect to claims 17-20, these claims depend from claim 16 and add further limitations that are neither anticipated nor made obvious by Kessler, Colavin, Miller, or a combination of these references. Accordingly, Applicant respectfully requests that the Examiner withdraw the rejections of claims 16-20.

CONCLUSIONS

Applicant believes this to be a complete response to all of the issues raised in the instant office action and further submits, in view of the amendments and arguments advanced above, that claims 1-25 are in condition for allowance. Reconsideration of the rejections is requested, and allowance of the claims is solicited.

Applicant also notes that any amendments made by way of this response, and the observations contained herein, are made solely for the purpose of expediting the patent application process in a manner consistent with the PTO's Patent business Goals (PBG), 65 Fed. Reg. 54603 (September 8, 2000), and are furthermore made without prejudice to Applicant under this or any other jurisdictions. It is moreover asserted that insofar as any subject matter might otherwise be regarded as having been abandoned or effectively disclaimed by virtue of amendments made herein and/or incorporated in attachments submitted with this response, Applicants wishes to reserve the right and hereby provides notice of intent to restore such subject matter and/or file a continuation application in respect thereof.

Applicant earnestly requests that the Examiner contact the undersigned practitioner by telephone if the Examiner has any questions or suggestions concerning this amendment, the application, or allowance of any claims thereof.

Respectfully submitted,
HUFFMAN PATENT GROUP, LLC

/Richard K. Huffman/

By: _____

RICHARD K. HUFFMAN, P.E.
Registration No. 41,082
Tel: (719) 575-9998

11/18/2008

Date: _____